INCA ASAP2 Description for CAN Monitoring

INCA User's Guide – Supplement

September 06, 2007

Copyright

Contents

1 Introduction

1.1 Definitions and Abbreviations

ASAP2

Description format. An ASAP2 description specifies a logical node as seen from the CAN.

BLOB

Binary large object: part of the ASAP2 description format.

Frame

A frame is a transmission sequence. Within this sequence signals are transmitted.

CAN

Controller Area Network: Standardized field bus.

CANdb Format (CANdb Description)

Certain description format of Vector Informatik GmbH. A CANdb description can specify multiple nodes.

Identifier

Individual bit sequence (11 or 29 bits) identifying a certain frame.

Signal

Information carried in a frame.

1.2 Conventions

The following typographical conventions are used in this document:

Choose **File** →**Open**.	Menu commands are shown in boldface.
Click **OK**.	Buttons are shown in boldface.
Press <ENTER>.	Keyboard commands are shown in angled brackets.
The "Open File" dialog box is displayed.	Names of program windows, dialog boxes, fields, etc. are shown in quotation marks.
Select the file `setup.exe`	Text in drop-down lists on the screen, program code, as well as path- and file names are shown in the Courier font.
A *distribution* is always a one-dimensional table of sample points.	General emphasis and new terms are set in italics.

2 Overview

Frames and signals occurring on the CAN bus are defined in a project description file.

In INCA, a project is used for the CAN monitoring functionality. This project can be loaded:

1. from a CANdb file, or
2. directly from an ASAP2 description.

2.1 Description using the CANdB Format

CANdb files have the file extension `*.dbc`. They have established themselves as a de-facto industry standard.

The *CANdb to ASAP2 converter* makes the use of the CANdb files very easy. The converter is implemented as a toolbox. Instead of an ASAP2 description (in the "Database browser" choose **Edit ->Add** and then **->ECU-Project (A2l)**), the user can load a project from the database by selecting **->CAN DB** in the **Edit ->Add**" menu.

After this project has been loaded, it cannot be distinguished anymore from a project that was loaded directly from an ASAP2 description. Handling of this project is identical in all parts of the tool.

2.2 Description using the ASAP2 Format

Apart from the CANdb format, the frames and signals can also be described directly in the ASAP2 format (since ASAP2 version 1.21).

The keywords FRAME, FRAME_MEASUREMENT and FUNCTION are used for this purpose. The description of the required blobs is explained in chapter 4, *BLOB Descriptions*, on page 10.

3 Description of the Frames and Signals in ASAP2

3.1 Signals

A CAN signal is described in ASAP2 by means of a measure variable (keyword MEASUREMENT).

Prototype

The ASAP2 description for a measure variable has the following structure:

```
/begin MEASUREMENT  /* Keyword */
   Name             /* Name */
   LongIdentifier   /* Description */
   Datatype         /* Data type */
   Conversion       /* Conversion formula */
   Resolution       /* Resolution */
   Accuracy         /* Accuracy */
   LowerLimit       /* Lower physical limit */
   UpperLimit       /* Upper physical limit */
   BYTE_ORDER ...   /* Byte order */
   BIT_MASK ...      /* Bit mask */
   IF_DATA ...      /* Device information */
   ...              /* Other information */
/end MEASUREMENT    /* End of keyword */
```

A detailed description of each statement can be found in the description of the ASAP2 format. Here there are no special features for CAN monitoring.

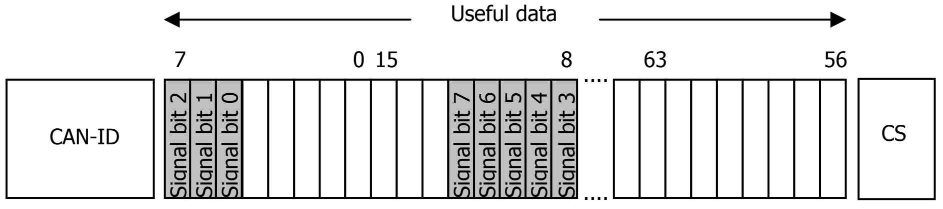
Example

A signal named aMeasurementName is contained in a CAN frame starting at bit 5 and has a length of 8 bits.

Message layout:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
-- 7	-- 6	--| 5	4	3	2	1	0	**byte 0**
15	14	13	<-- 12	-- 11	-- 10	-- 9	-- 8	**byte 1**
23	22	21	20	19	18	17	16	**byte 2**
31	30	29	28	27	26	25	24	**byte 3**
39	38	37	36	35	34	33	32	**byte 4**
47	46	45	44	43	42	41	40	**byte 5**
55	54	53	52	51	50	49	48	**byte 6**
63	62	61	60	59	58	57	56	**byte 7**

CAN message:



This signal is defined in an ASAP2 description as follows:

```
/begin MEASUREMENT                /* Keyword */
   aMeasurementName               /* Name */
   "description"                  /* Description */
   UBYTE                          /* Data type */
   aFormulaForMeasurment          /* Conversion formula */
   2                              /* Resolution */
   2.5                            /* Accuracy */
   120.0                          /* Lower physical limit */
   8400.0                         /* Upper physical limit */
   BYTEORDER MSB_LAST             /* Byte order */
   BIT_MASK 0xFF                  /* Bit mask */
   /begin IF_DATA CAN_MONITORING /* Device information */
      /begin KP_BLOB
         0x5                      /* Start bit*/
         8                        /* Length */
      /end KP_BLOB
   /end IF_DATA
/end MEASUREMENT                  /* End of keyword */
```

A detailed description of the IF_DATA section can be found in chapter 4, *BLOB Descriptions*, on page 10.

3.2 Frames

3.2.1 Physical Collection of Signals

Signals are sent in frames on the CAN bus. Each frame can contain several signals.

This combination of signals in frames is referred to as a physical collection. To describe this collection in ASAP2, the keywords FRAME and FRAME_MEASUREMENT are used.

The physical description is mandatory. Otherwise, signals cannot be measured on the CAN bus.

Prototype

This is the description pattern of a frame:

```
/begin FRAME                   /* Keyword */
   Name                        /* Name */
   LongIdentifier              /* Description */
   ScalingUnit                 /* Scaling of the grid */
   Rate                        /* Grid */
   FRAME_MEASUREMENT ...       /* Signals */
   IF_DATA ...                 /* Device information */
/end FRAME                     /* End of keyword */
```

A detailed description of each statement can be found in the description of the ASAP2 format. Here there are no special features for CAN monitoring.

Example

A frame named aFrameName has the CAN ID 0x200 (11-bit identifier) and a length of 8 bytes.

```
/begin FRAME                           /* Keyword */
   aFrameName                          /* Name */
   "description of frame"              /* Description */
   0                                   /* Scaling of the grid */
   0                                   /* Grid */
   FRAME_MEASUREMENT aMeasurementName /* Signals */
   /begin IF_DATA CAN_MONITORING       /* Device information */
      QP_BLOB 0x200 0 8                /* Id, Id length, frame length */
   /end IF_DATA
/end FRAME                             /* End of keyword */
```

A detailed description of the IF_DATA section can be found in chapter 4, *BLOB Descriptions*, on page 10.

3.2.2 Logical Collection of Signals

In addition to the physical collection (signals in a frame), it is also possible to define a logical (orthogonal) collection of signals. This allows collecting all signals related to, for example, engine control.

The logical collection of signals is optional.

Normally the logical collection is chosen so that it matches the physical collection. This has the advantage, among others, that the signals in the variable selection can be displayed sorted by frames.

Prototype

The logical collection is described using functions. The pattern for the description of functions is as follows:

```
/begin FUNCTION            /* Keyword */
   Name                    /* Name */
   LongIdentifier          /* Description */
   OUT_MEASUREMENT ...     /* List of signals */
   ...
/end FUNCTION              /* End of keyword */
```

A detailed description of each statement can be found in the description of the ASAP2 format. Here there are no special features for CAN monitoring.

Example

A measure variable named aMeasurementName shall be indicated in a function named aFunctionName, since the CAN frame aFunctionName contains the signal aMeasurementName. This means that the logical view is the same as the physical view (collection of signals in frames).

```
/begin FUNCTION                /* Keyword */
   aFunctionName               /* Name */
   "description of function"   /* Description */
   /begin OUT_MEASUREMENT      /* List of signals */
      aMeasurementName
   /end OUT_MEASUREMENT
/end FUNCTION                  /* End of keyword */
```

The statement OUT_MEASUREMENT aMeasurementName requires that a description of a measure variable (MEASUREMENT) exists with the name aMeasurementName (name-based reference).

4 BLOB Descriptions

The following three types of BLOBs are used for CAN monitoring:

- TP_BLOB
- QP_BLOBs with the FRAME keyword
- KP_BLOBs with the MEASUREMENT keyword.

BLOB descriptions are always included in the IF_DATA section of the corresponding keyword. CAN monitoring always evaluates the IF_DATA section named CAN_MONITORING.

4.1 TP_Blob

4.1.1 Description

The TP_Blob defines the a1b driver specific parameters.

For CAN monitoring only the CAN baudrate is part of the TP_Blob.

Note

The CANdb does not contain any information about the CAN bus settings.

An asap2 description which was generated by INCA from a CANdb file contains the baudrate value which was defined via the appropriated setting in the "Hardware Configuration Window".

4.1.2 Example

CAN monitoring should be done on a CAN bus with a baudrate of 500 kBaud.

```
/begin IF_DATA CAN_MONITORING
   /begin  TP_BLOB
      500           /* Baudrate in kBaud */
   /end  TP_BLOB
/end IF_DATA
```

4.1.3 AML Description

For a more detailed description of BLOBs, ASAP2 uses the so-called ASAP2 Meta Language (AML):

```
taggedunion IF_DATA
{
   "CAN_MONITORING" taggedstruct
   {
      block „TP_BLOB“ struct
      {
         long;      /* Baudrate in kBaud */
      };
   };
};
```

For a description of the AML syntax, refer to "*ASAP2 Interface Specification*" [1], Chapter 8.

4.2 QP_BLOB

4.2.1 Description

A QP_BLOB is used for the description of the CAN-specific parameters at the frame level (defined in ASAP2 with the FRAME keyword).

The specific parameters at the frame level are the frame ID (CAN-ID), the ID length (11- or 27-bit identifiers), and the frame length.

Frames without signals and QP_BLOB are ignored.

4.2.2 Example

A frame named aFrameName has the frame ID 0x0123. It uses a 27-bit identifier. The frame contains the two signals aMeasurementName1 and aMeasurementName2. The comment is for information only.

```
/begin FRAME                    /* Keyword */
   aFrameName                   /* Name */
   "description of frame"       /* Description */
   0                            /* Scaling of the grid */
   0                            /* Grid */
   FRAME_MEASUREMENT            /* List of signals */
      aMeasurementName1
      aMeasurementName2
   /begin IF_DATA CAN_MONITORING /* Device information */
      QP_BLOB 0x0123 1 5        /* Id, Id length, frame length */
   /end IF_DATA
/end FRAME                      /* End of keyword */
```

Definition of ID:

The identifier (CAN-ID) of the frame must be specified as a hexadecimal value.

Definition of ID length:

The CAN protocol allows two different lengths for the frame identifier:

- For 11-bit identifiers: ID Length bit = 0
- For 27-bit identifiers (extended CAN): ID Length bit = 1

Definition of frame length:

The length of the CAN frame is specified in bytes allowing values from 1 to 8.

4.2.3 AML Description

For a more detailed description of BLOBs, ASAP2 uses the so-called ASAP2 Meta Language (AML):

```
taggedunion IF_DATA
{
   "CAN_MONITORING" taggedstruct
   {
      block „QP_BLOB“ struct
      {
         long;                /* Id */
         char;                /* Id length */
         char;                /* Frame length */
      };
   };
};
```

For a description of the AML syntax, refer to "*ASAP2 Interface Specification*" [1], Chapter 8.

4.3 KP_BLOB

4.3.1 Description

Each signal occurring in a CAN frame must be described in ASAP2 as a measure variable (MEASUREMENT).

The CAN-specific extensions are contained in the KP_BLOB. They are linked to the frames via the frame name.

MULTIPLEX is supported (optional). In multiplex mode, different messages (signals) are transmitted in one frame. These messages can be distinguished in the data section of the message by means of the mode signal.

Multiplex signals and regular signals can in principle also be mixed within a frame.

4.3.2 Specific Byte Order

Both the MOTOROLA byte order and the INTEL byte order can be used. The byte order is specified individually for each signal, i.e., a frame may contain signals having different byte orders.

When using the INTEL byte order (in ASAP2 specified by MSB_LAST), the start bit defines the lsb (last significant bit) of the LSB (Last Significant Byte) of the signal.

When using the MOTOROLA byte order (in ASAP2 specified by MSB_FIRST), the start bit defines the msb (most significant bit) of the MSB (Most Significant Byte) of the signal.

Note

This is different to the definition of the start bit within the CANdb. There the start bit always defines the lsb of the LSB (for both byte order definitions).

Changing the byte order therefore requires a change in the start bit.

Note

The bit order within a byte is the same for both byte order definitions! Bit 0 always represents lsb and bit 7 always represents msb of a byte.

Example

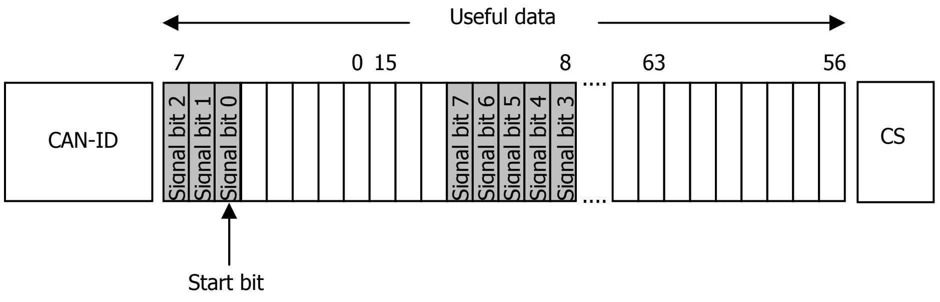
A signal named aMeasurementName has a length of 8 bit but is transmitted over 2 bytes and is defined in INTEL format.

Its start bit is bit 5.

Message layout:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
<- 7 -	- 6 -	5 (lsb)	4	3	2	1	0	**byte 0** (LSB)
15	14	13	12 (msb)	- 11 -	- 10 -	- 9 -	- 8 -	**byte 1** (MSB)
23	22	21	20	19	18	17	16	**byte 2**
31	30	29	28	27	26	25	24	**byte 3**
39	38	37	36	35	34	33	32	**byte 4**
47	46	45	44	43	42	41	40	**byte 5**
55	54	53	52	51	50	49	48	**byte 6**
63	62	61	60	59	58	57	56	**byte 7**

CAN message:



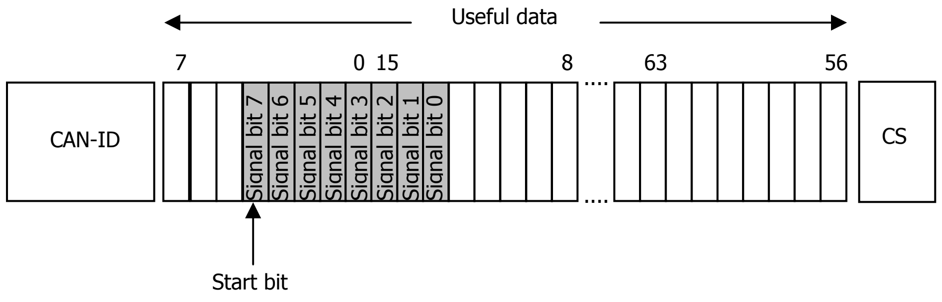
A signal named aMeasurementName1 has a length of 8 bit and is defined in MOTOROLA format.

Its start bit is bit 4.

Message layout:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
7	6	5	4 (msb)	- 3 -	- 2 -	- 1 -	- 0 -	**byte 0** (MSB)
<- 15 -	- 14 -	13 (lsb)	12	11	10	9	8	**byte 1** (LSB)
23	22	21	20	19	18	17	16	**byte 2**
31	30	29	28	27	26	25	24	**byte 3**
39	38	37	36	35	34	33	32	**byte 4**
47	46	45	44	43	42	41	40	**byte 5**
55	54	53	52	51	50	49	48	**byte 6**
63	62	61	60	59	58	57	56	**byte 7**

CAN message:



4.3.3 Examples

Non-Multiplexed Signal

The signal aMeasurementName is located in its frame starting at bit 5 and has a length of 8 bits.

The signal aMeasurementName is stored in the frame using the INTEL format and is interpreted as a signed variable.

```
/begin MEASUREMENT                  /* Keyword */
   aMeasurementName                 /* Name */
   "description of measurement"     /* Description */
   SBYTE                            /* Data type */
   aConversion                      /* Conversion formula */
   2                                /* Resolution */
   2.5                              /* Accuracy */
   120.0                            /* Lower physical limit */
   8400.0                           /* Upper physical limit */
   BYTEORDER MSB_LAST               /* Byte order */
   BIT_MASK 0xFF                    /* Bit mask */
   /begin IF_DATA CAN_MONITORING    /* Device information */
      /begin KP_BLOB 0x5 8          /* Start bit, length */
      /end KP_BLOB
   /end IF_DATA
/end MEASUREMENT                    /* End of keyword */
```

Definition of start bit:

The start bit is specified as a hexadecimal number.

Definition of length:

The number of bits describing the signal can be 1 to 32. The maximum number of bits depends on the data type of the measure variable.

The signal aMeasurementName is located in its frame starting at bit 42 (corresponds to 0x2A in hexadecimal notation) and has a length of 8 bits.

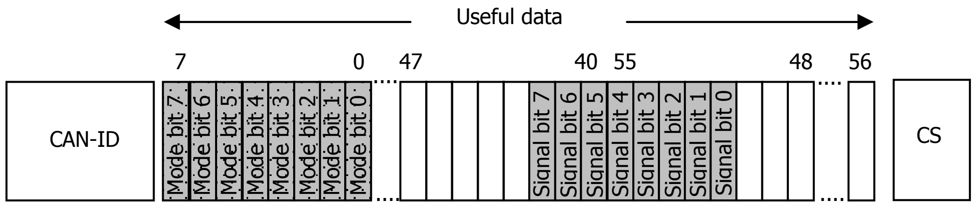
However, because it is a mode-dependent signal, this is the case only if the 8 bits starting at bit 0 (interpreted as unsigned word size in the Motorola format) have the value 7.

The mode-dependent signal aMeasurementName is stored in the frame using the Motorola format and is interpreted as an unsigned variable.

Message layout:

bit 7	**bit 6**	**bit 5**	**bit 4**	**bit 3**	**bit 2**	**bit 1**	**bit 0**	
7 (msb)	- 6 -	- 5 -	- 4 -	- 3 -	- 2 -	- 1 -	0 (lsb)	**byte 0**
15	14	13	12	11	10	9	8	**byte 1**
23	22	21	20	19	18	17	16	**byte 2**
31	30	29	28	27	26	25	24	**byte 3**
39	38	37	36	35	34	33	32	**byte 4**
47	46	45	44	43	42 (msb)	- 41 -	- 40 -	**byte 5** (MSB)
<- 55 -	- 54 -	- 53 -	- 52 -	51 (lsb)	50	49	48	**byte 6** (LSB)
63	62	61	60	59	58	57	56	**byte 7**

CAN message:



Note

The mode signal is also called multiplexor.

The mode-dependent signal is also called multiplexed signal.

```
/begin MEASUREMENT                  /* Keyword */
   aMeasurementName                 /* Name */
   "description of measurement"     /* Description */
   UBYTE                            /* Data type */
   AConversion                      /* Conversion formula */
   2                                /* Resolution */
   2.5                              /* Accuracy */
   120.0                            /* Lower physical limit */
   8400.0                           /* Upper physical limit */
   BYTEORDER MSB_FIRST              /* Byte order */
   BIT_MASK 0xFF                    /* Bit mask */
   /begin IF_DATA CAN_MONITORING    /* Device information */
      /begin KP_BLOB 0x2A 8         /* Startbit, Length */
         MULTIPLEX                  /* Multiplex information */
         7                          /* Start bit */
         8                          /* Length */
         SBYTE                      /* Data type */
         MSB_FIRST                  /* Byte order */
         7                          /* Multiplexor/Mode value */
      /end KP_BLOB
   /end IF_DATA
/end MEASUREMENT                    /* End of keyword */
```

Signals that are not defined as multiplexers are treated as if they were defined as multiplex by MULTIPLEX 1 0 UWORD MSB_FIRST 0. This allows implementing a common interface for multiplexed and non-multiplexed signals to the ASAP1B driver. The multiplex section in the description is then simply unnecessary.

4.3.4 AML Description

For a more detailed description of BLOBs, ASAP2 uses the so-called ASAP2 Meta Language (AML):

```
taggedunion IF_DATA
{
   "CAN_MONITORING" taggedstruct
   {
      block „KP_BLOB“ struct
      {
         char;                      /* Start bit */
         char;                      /* Length */
         taggedstruct
         {
           “MULTIPLEX” struct       /* Multiplexer information */
            {
               char;                /* Start bit */
               char;                /* Length */
               char[];              /* Data type */
               char[];              /* Byte order */
               long;                /* Value */
            };
         };
      };
   };
};
```

For a description of the AML syntax, refer to "*ASAP2 Interface Specification*" [1], Chapter 8.

5 Appendix

5.1 AML Description

```
/********************************************************************/
/*                                                                  */
/*   ASAP2 Meta Language for CAN-Monitoring V1.1                    */
/*   Assumes ASAP2 V1.121                                           */
/*                                                                  */
/*   ETAS GmbH                                                      */
/*                                                                  */
/*   Datatypes:                                                     */
/*                                                                  */
/*   A2ML       ASAP2          Windows  Erlaeuuterung               */
/*   -------------------------------------------------------        */
/*   uchar      UBYTE          BYTE     unsigned 8 Bit              */
/*   char       SBYTE          char     signed 8 Bit                */
/*   uint       UWORD          WORD     unsigned integer 16 Bit     */
/*   int        SWORD          int      signed integer 16 Bit       */
/*   ulong      ULONG          DWORD    unsigned integer 32 Bit     */
/*   long       SLONG          LONG     signed integer 32 Bit       */
/*   float      FLOAT32_IEEE            float 32 Bit                */
/*                                                                  */
/********************************************************************/


taggedunion IF_DATA
{
   "CAN_MONITORING" taggedstruct
   {
      block „TP_BLOB“ struct
      {
         ulong;                /* Baudrate in kBaud */
      };
   };
};



taggedunion IF_DATA
{
   "CAN_MONITORING" taggedstruct
   {
      block "QP_BLOB" struct
      {
         long;                /* Message identifier */
         char;                /* Identifier length */
         char;                /* Message length */
      };
   };
};
```

```
taggedunion IF_DATA
{
   "CAN_MONITORING" taggedstruct
   {
      block "KP_BLOB" struct
      {
         char;                /* Start bit */
         char;                /* Number of bits */
         taggedstruct
         {
            "MULTIPLEX" struct
            {
               char;          /* Start bit */
               char;          /* Number of bits */
               char;          /* Type and sign */
               char;          /* Byte order */
               long;          /* Tag */
            };
         };
      };
   };
};
```

5.2 ASAP2 Description

```
ASAP2_VERSION 1 30
   /begin PROJECT
      aProjectName
      "description of project"

      /begin HEADER
         "project"
         VERSION "1.0"
         PROJECT_NO "1.0"
      /end HEADER

      /begin MODULE
         aModuleName
         "description of module"

         /begin MOD_PAR
            ""
         /end MOD_PAR

         /begin IF_DATA CAN_MONITORING
            /begin  TP_BLOB
               500
           /end  TP_BLOB
         /end IF_DATA

         /begin MEASUREMENT
            aMeasurementName
            "description of measurement"
            ULONG
            aConversionName
            0
            0.0
            0
            1000
            /begin IF_DATA CAN_MONITORING
               /begin KP_BLOB
                  0x0 32
               /end KP_BLOB
            /end IF_DATA
            FORMAT ""
            BYTE_ORDER MSB_LAST
            BIT_MASK 0xFFFFFFFF
         /end MEASUREMENT

         /begin COMPU_METHOD
            aConversionName
            "description of conversion"
            RAT_FUNC
            "%f5.2"
            ""
            COEFFS 0 1.0 0.0 0 0 1
         /end COMPU_METHOD
         /begin FRAME
            aFrameName
            "description of frame"
            0
            0
            /begin IF_DATA CAN_MONITORING
               QP_BLOB 0x0200 0 8
            /end IF_DATA
            FRAME_MEASUREMENT aMeasurementName
         /end FRAME

         /begin FUNCTION
            aFunctionName
            "description of function"
            /begin OUT_MEASUREMENT
               aMeasurementName
            /end OUT_MEASUREMENT
         /end FUNCTION

      /end MODULE

   /end PROJECT
```

6 Referenced Documents

[1] ASAP2 Interface Specification – Interface 2 (ASAM MCD 2MC / ASAP2)
Version 1.51, Release 2003-03-11
http://www.asam.net/03_standards_06.php

7 ETAS Contact Addresses

Europe (except France, Belgium, Luxembourg and Great Britain)

ETAS GmbH

Borsigstraße 14	Phone:	+49 711 89661-0
70469 Stuttgart	Fax:	+49 711 89661-105
Germany	E-mail:	sales@etas.de
	WWW:	www.etasgroup.com

France, Belgium, Luxembourg

ETAS S.A.S.

1, place des Etats-Unis	Phone:	+33 1 56 70 00 50
SILIC 307	Fax:	+33 1 56 70 00 51
94588 Rungis Cedex	E-mail:	sales@etas.fr
France	WWW:	www.etasgroup.com

Great Britain

ETAS Ltd.

Studio 3, Waterside Court	Phone:	+44 1283 54 65 12
Third Avenue, Centrum 100	Fax:	+44 1283 54 87 67
Burton-upon-Trent	E-mail:	sales@etas-uk.net
Staffordshire DE14 2WQ	WWW:	www.etasgroup.com
Great Britain		

USA

ETAS Inc.

3021 Miller Road	Phone:	+1 888 ETAS INC
Ann Arbor, MI 48103	Fax:	+1 734 997-9449
USA	E-mail:	sales@etas.us
	WWW:	www.etasgroup.com

Japan

ETAS K.K.

Queen's Tower C-17F	Phone:	+81 45 222-0900
2-3-5, Minatomirai, Nishi-ku	Fax:	+81 45 222-0956
Yokohama 220-6217	E-mail:	sales@etas.co.jp
Japan	WWW:	www.etasgroup.com

Korea

ETAS Korea Co. Ltd.

4F, 705 Bldg. 70-5	Phone:	+82 2 57 47-016
Yangjae-dong, Seocho-gu	Fax:	+82 2 57 47-120
Seoul 137-889	E-mail:	sales@etas.co.kr
Korea		

P.R. China

ETAS (Shanghai) Co., Ltd.

2404 Bank of China Tower	Phone:	+86 21 5037 2220
200 Yincheng Road Central	Fax:	+86 21 5037 2221
Shanghai 200120, P.R. China	E-mail:	sales.cn@etasgroup.co m
	WWW:	www.etasgroup.com